Motion Imagery Standards Board **Engineering Guideline:** **Remote Video Terminal Local Data Set**	**MISB EG 0806.1** **18 December 2008**

1 Scope

This Engineering Guideline defines a Local Data Set (LDS) that may be used with a Remote Video Terminal (RVT), lays out the relationship between the RVT LDS and other relevant Standards, and gives implementation guidance for the RVT LDS.

2 References

This Engineering Guideline references the following documents and standards:

2.1 Normative References

DMA TM8358.1: Datums, Ellipsoids, Grids, and Grid Reference Systems, 20 September 1990.

IEEE POSIX Standard IEEE 1003.

ISO/IEC 1318-1:2000.

MIL-STD-2525B: Common Warfighting Symbology, 1 July 2005.

MISB Standard 0807, 18 September 2008.

MISB Standard 0807.1, 9 December 2008

SMPTE 336M-2007: Data Encoding Protocol Using Key-Length-Value.

SMPTE RP 210.11: KLV Metadata Dictionary.

2.2 Informative References

MISB EG 0104.5: Predator UAV Basic Universal Metadata Set.

MISB Standard 0601.2: UAS Datalink Local Data Set.

MISB Standard 0102.5: Security Metadata Universal and Local Data Sets for Digital Motion Imagery.

MISB RP 0103.1: Timing Reconciliation Universal Metadata Set for Digital Motion Imagery.

MISB Standard 0604: Time Stamping Compressed Motion Imagery.

3 Introduction

MISB Standard 0601.2 is proving to be highly successful and is being adopted by a wide range of users. As is generally the case with a successful standard, new adopters often wish to extend it while current users wish it to remain stable.

The ROVER (Remotely Operated Video Enhanced Receiver) program is attracted to the functionality of Standard 0601.2, but requires additional metadata elements unique to its

customers' missions. The purpose of this Engineering Guideline is to formalize a method of communicating with a Remote Video Terminal (ROVER or other related programs, such as the One System Remote Video Terminal (OSRVT)) and create a configuration-managed metadata standard to meet its needs with minimal impacts to Standard 0601.2 users.

The RVT Local Data Set (LDS) defined in this Engineering Guideline can stand alone as its own local data set or be embedded within other metadata sets (like Standard 0601.2). This provides the ability for system designers to produce or receive one, the other, or both metadata standards based upon program requirements.

4 LDS Packet Structure

Local Data Sets are more bit-efficient than individual keys but allow for a great deal of flexibility in implementation, allowing users to tailor implementations to their specific and changing needs. This section describes how to create a Local Data Set in accordance with SMPTE 336-M.

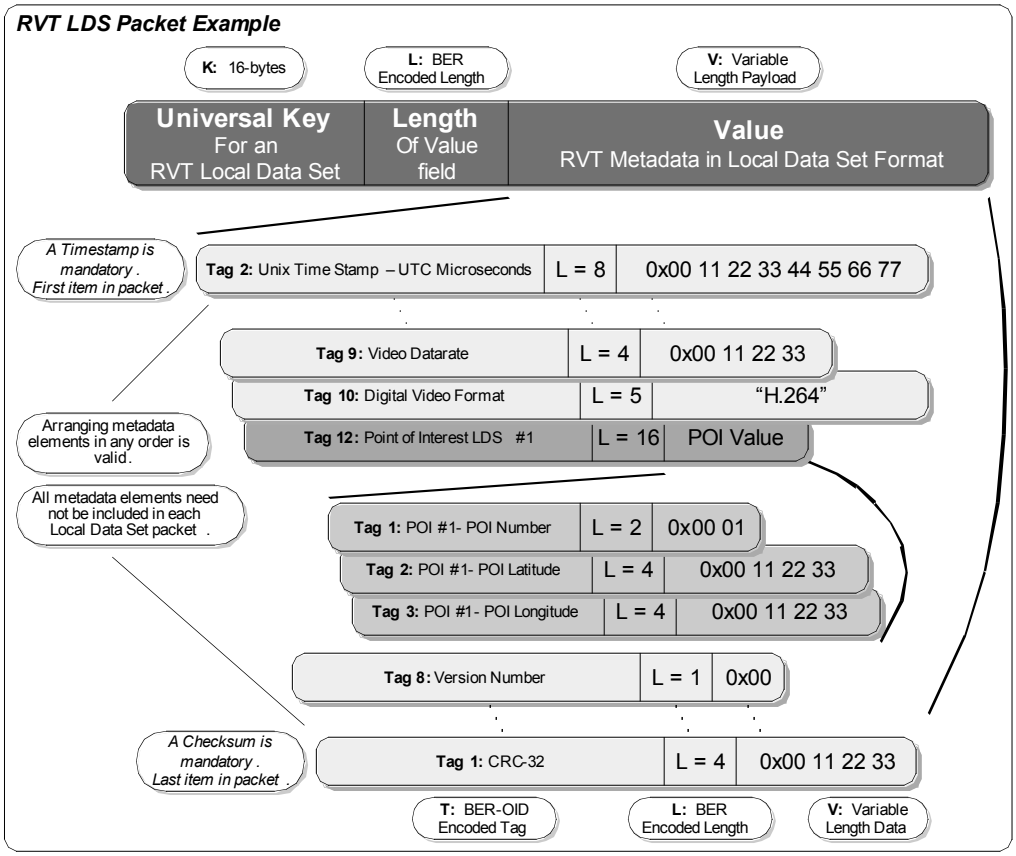

Figure 4-1: Example Local Data Set Packet

Figure 4-1 shows the general format of how the RVT LDS is configured. It is required that each LDS packet contain a Unix-based Coordinated Universal Time (UTC) timestamp that represents the time of birth of the metadata within the LDS packet. A checksum metadata item is also required to be included in each LDS packet.

Any combination of metadata items can be included in a RVT Local Data Set packet. The items within the RVT LDS can be arranged in any order except that the Unix-based UTC Time Stamp must come first and the Checksum must come last.

4.1 Bit and Byte Ordering

All metadata is represented using big-endian (Most Significant Byte (MSB) first) encoding. Bytes are big-endian bit encoding (most significant bit (msb) first).

4.2 Tag and Length Field Encoding

Both the LDS metadata item Tag and length fields are encoded using basic encoding rules (BER) for either short or long form encoding of octets. This length encoding method provides the greatest level of flexibility for variable length data contained within a KLV packet.

In practice, the majority of metadata items in a LDS packet will use the short form of key and length encoding which requires only a single byte to represent the length. The length of the entire LDS packet, however, is often represented using the long form of length encoding since the majority of packets have a payload larger than 127 bytes. The key for the entire LDS packet is always 16 bytes. The length of a single packet is represented by 2 bytes whenever the payload portion of the LDS packet is larger than 127, but less than 256 bytes. Both short and long form encoding is discussed in the subsections that follow.

See SMPTE 336M Section 3.2 for further details.

4.2.1 BER Short Form Length Encoding Example

For LDS packets and data elements shorter than 128 bytes, the length field is encoded using the BER short form. Length fields using the short form are represented using a single byte (8 bits). The most significant bit in this byte signals that the long form is being used. The last seven bits depict the number of bytes that follow the BER encoded length. An example LDS packet using a short form encoded length is shown below:

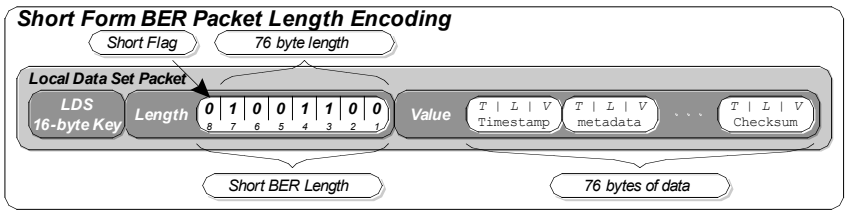

Figure 4-2: Example Short Form Length Encoding

Although this example depicts the length field of the entire LDS packet, short form BER encoding also applies to the keys and lengths within the LDS packet.

4.2.2 BER Long Form Length Encoding

For LDS packets and data elements longer than 127 bytes, the length field is encoded using the BER long form. The long form encodes length fields using multiple bytes. The first byte indicates long form encoding as well as the number of subsequent bytes that represent the length. The bytes that follow the leading byte are the encoding of an unsigned binary integer equal to the number of bytes in the packet. An example LDS packet using a long form encoded length is shown in Figure 4-3.

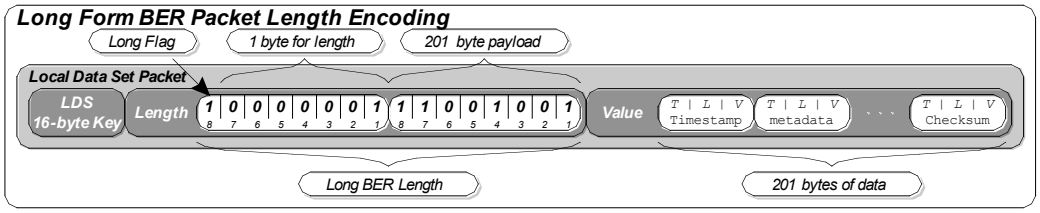

Figure 4-3: Example Long For Length Encoding

Although this example depicts long form BER encoding on the length field of the entire LDS packet, long form BER encoding also applies to the tags and lengths within the LDS packet.

4.3 Time Stamping

Every RVT LDS KLV packet is required to include a Unix-based UTC timestamp as a way to correspond the metadata with a standardized time reference. UTC time is useful to associate metadata with frames, and for reviewing time-critical events at a later date. This section describes how to include a timestamp within a Local Data Set packet.

Metadata sources are coordinated to operate on the same standard time, which is typically GPS derived. The metadata source provides a timestamp for inclusion in a LDS packet and the timestamp assists the accuracy of synchronizing each frame to its corresponding metadata set.

The mandatory timestamp tag is User Defined Time Stamp – Microseconds Since 1970. The UTC timestamp (RVT LDS Tag 02) is an 8 byte unsigned integer that represents the number of microseconds that have elapsed since midnight (00:00:00), January 1, 1970. This date is known as the UNIX Epoch and is discussed in the IEEE POSIX standard IEEE 1003.1.

4.4 Packet Timestamp

The Packet Timestamp is inserted at the beginning of the value portion of a RVT LDS KLV packet. The timestamp is represented by RVT LDS Tag 02 (see above) and applies to all metadata in the LDS packet. This timestamp corresponds to the time of birth of all the data within the LDS packet. This time can be used to associate the metadata with a particular video frame and be displayed or monitored appropriately.

An example LDS packet containing a timestamp is show below:

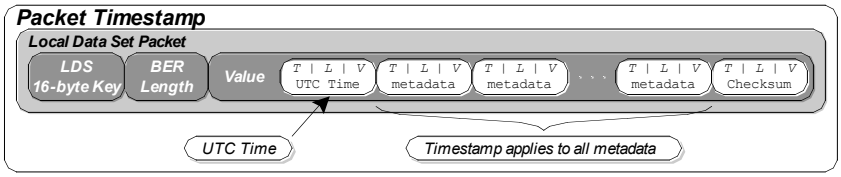

Figure 4-4: Example Packet Timestamp

4.5 Error Detection

To help prevent erroneous metadata from being presented with video, it is required that a 32-bit cyclic redundancy check (CRC) be included in every RVT Local Data Set instance. The CRC, given by RVT LDS Tag 01 is the CRC-32 checksum defined in ISO/IEC 13818-1:2000 (MPEG2), and must be located at the end of each instance of the RVT LDS. The CRC is a

running 32-byte sum through the entire LDS packet starting with the 16 byte Local Data Set key and ending with summing the length field of the checksum data item.

Note that this CRC-32 differs from the one commonly used in IP applications.

The figure below shows the data range that the CRC is performed over:

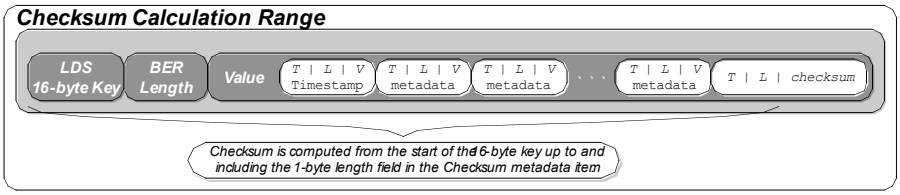

Figure 4-5: Example Checksum Computation Range

An example algorithm for calculating the checksum is given in Section 8.

4.6 Key to Tag Mappings

It is required that tags within a Local Data Set map back to valid SMPTE RP 210 (or MISB Standard 0807) keys. When a similar but not identical entry already exists in SMPTE RP210 (or MISB Standard 0807), a new key should be assigned, preferably as a sub-leaf of the existing entry (*i.e.* replacing the first zero byte of the existing key with 01 or the next available number). When it is not possible to assign a new key, the differences shall be noted in SMPTE RP2009 Groups Register (or the equivalent MISB Standard 0807) and the entry in RP210 shall be informatively noted as Context-Dependent.

As an example, the Video Frame Counter Tag is a three-byte integer representation of the Frame Code Key, which is a 31-character string. There is no loss of information in moving a 3-byte unsigned integer into a 31-character string.

It is also possible to map multiple Tags back to a single Key (*e. g.* the E0 – E8 series Tags) because each instantiation within the LDS carries a unique Tag and is therefore an unambiguous reference.

5 RVT Local Data Set Conventions

This section defines the RVT Local Data Set (LDS). The tags that are supported in this LDS are defined and mapped to metadata items in the SMPTE KLV Metadata Dictionary (SMPTE RP-210.11) or in MISB Standard 0807. The RVT LDS is SMPTE 336M-2007 compliant.

5.1 RVT LDS Universal Keys

The sixteen-byte Universal Key to be used with the RVT LDS is:

RVT Local Data Set Key (hex): **06 0E 2B 34 02 0B 01 01 0E 01 03 01 02 00 00 00** Release Date: 22 April 2008 Release Version: MISB Standard 0807

Please note that the earlier version of the RVT LDS Universal Key (included here for historical reference) is not within the MISB Metadata Annex domain and was never registered with SMPTE and, therefore, should not be used.

The elements of the RVT LDS are defined in Table 6-1. Subordinate data structures for the RVT LDS are described in section 5.5 with element listings listed in Table 6-2, Table 6-3, and

Table 6-4.

ROVER Local Data Set (*sic*) Key (hex): **06 0E 2B 34 01 01 01 01 0F 4C 33 43 53 57 01 00** Release Date: **NOT APPROVED** Release Version: **NOT APPROVED; INCLUDED FOR HISTORICAL REFERNCE ONLY**

5.2 RVT LDS Tag Formats and Lengths

Tag lengths for the RVT LDS are BER-OID encoded as indicated by byte 7 in the 16-byte key in accordance with SMPTE 336m-2007.

Tag numbers listed within this document are all in decimal unless preceded by "0x" or "0b" for hexadecimal or binary (respectively).

5.3 RVT LDS Required Tags

All standalone instantiations of the RVT LDS shall contain the CRC-32 (RVT LDS Tag 01) and the UNIX-based UTC Timestamp (RVT LDS Tag 02).

Implementations nesting the RVT LDS within another Local Set can optionally exclude including the RVT LDS CRC-32 and UTC timestamp when the external LDS include a timestamp and checksum.

5.4 RVT LDS Duplication of Tags

With the exception of the RVT LDS Subordinate Set tags, metadata items within an RVT LDS Packet cannot be represented multiple times.

For instance, tag 03 representing Platform True Airspeed can only appear once in an RVT LDS packet whereas tag 12, representing a Point of Interest, can appear multiple times to convey information for multiple points of interest.

5.5 RVT LDS Subordinate Sets

The RVT LDS makes use of three smaller Local Data Sets. These embedded local sets are:

1. Point of Interest (POI) Local Data Set.
2. Area of Interest (AOI) Local Data Set.
3. User Defined Data Local Data Set.

5.5.1 Point of Interest Local Data Set

Point of Interest Local Data Set Key (hex): **06 0E 2B 34 02 0B 01 01 0E 01 03 01 0C 00 00 00** Release Date: 31 July 2008 Release Version: MISB Standard 0807

The elements of the Point of Interest Local Data Set (POI LDS) are defined in Table 6-2.

Note that all instantiations of the POI LDS are required to contain (at a minimum) the POI/AOI Number (Tag 01), POI Latitude (Tag 02), and POI Longitude (Tag 03). Items with tags 04 through 09 are optional.

No items within a POI LDS packet are allowed to appear multiple times within the same packet.

5.5.2 Area of Interest Local Data Set

Area of Interest Local Data Set Key (hex): **06 0E 2B 34 02 0B 01 01 0E 01 03 01 0D 00 00 00** Release Date: 31 July 2008 Release Version: MISB Standard 0807

The elements of the Area of Interest Local Data Set (AOI LDS) are defined in Table 6-3.

Note that all instantiations of the AOI LDS are required to contain (at a minimum) the POI/AOI Number (Item 01), Upper Left Latitude (Item 02), Upper Left Longitude (Item 03), Lower Right Latitude (Item 04), Lower Right Longitude (Item 05), and Type (Item 06). Items with tags 07 through 09 are optional.

No items within an AOI LDS packet are allowed to appear multiple times within the same packet.

5.5.3 User Defined Data Local Data Set

User Defined Data Local Data Set Key (Hex): **06 0E 2B 34 02 0B 01 01 0E 01 03 01 0F 00 00 00** Release Date: 9 December 2008 Release Version: MISB Standard 0807.1

The elements of the User Defined Data Local Data Set are defined in

Table 6-4

The data items contained in the User Defined Data LDS can only be mapped to SMPTE Experimental (Class 15) keys and cannot be mapped to specific SMPTE RP 210 or MISB Standard 0807 Keys. The keys provided in this document solely for the sake of completeness. All instances of the User Defined LDS must contain as their first element the Numeric ID for Data Type (Item 1) and User Data (Item 2) as their second element.

No other elements can be included in a User Defined Data LDS.

No items within the User Defined Data LDS packet are allowed to appear multiple times within the same packet.

5.5.4 Subordinate Set Example

Figure 5-1 depicts an example RVT LOS packet containing two Point of Interest LDS packets.

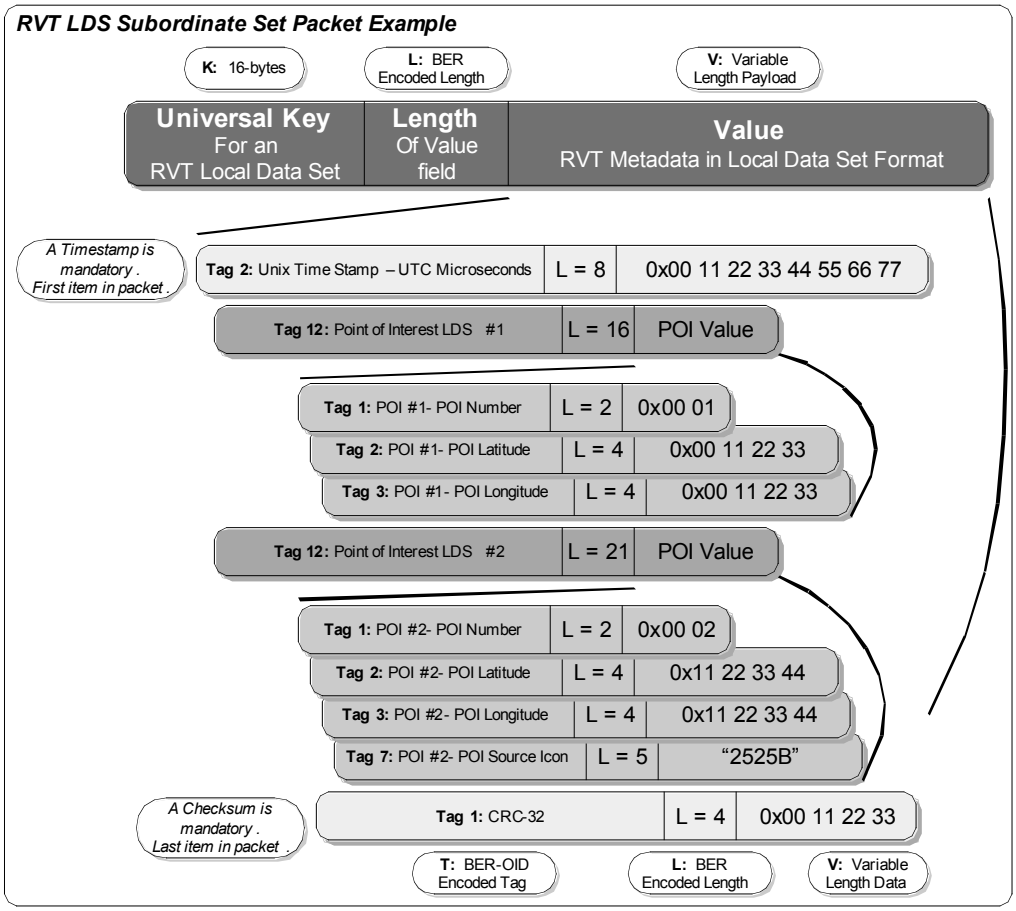

Figure 5-1: Example RVT Subordinate Set Packet

Note how both POI LDS packets contain all mandatory items, and POI 2 adds information for MIL-STD-2525B symbology. Also note that the RVT LDS packet contains the mandatory timestamp and checksum metadata items.

The other Subordinate Sets closely mimic the RVT LDS packet structure shown in Figure 5-1 according to the guidelines portrayed in section 5.5.

6 RVT Local Data Set Tables

Table 6-1: RVT Local Data Set

RVT LDS						
Key Value (hex)		Key Name	Units	Format	Length in Bytes	Notes
06 0E 2B 34 02 0B 01 01 0E 01 03 01 02 00 00 00		RVT KLV Dictionary	None	N/A	Variable	This is the Universal Key for the RVT LDS
Tag ID	Key Value (hex)	Key Name	Units	Format	Length in Bytes	Notes
01	06 0E 2B 34 01 01 01 01 0E 01 02 03 10 00 00 00	CRC 32	None	uint32	4	The checksum is per the CRC-32-ISO/IEC 13818-1:2000. Performed on entire LDS packet, including 16-byte UDS key. Note: This is Not the same Checksum as is used in EG0601. This checksum must appear as that last item in an RVT LDS pack when used.
02	06 0E 2B 34 01 01 01 03 07 02 01 01 01 05 00 00	User Defined Time Stamp - Microseconds Since 1970	Micro-seconds	uint64	8	Represents the Coordinated Universal Time (UTC) in Microseconds elapsed since midnight (00:00:00), January 1, 1970 (the UNIX Epoch). Derived from the POSIX IEEE1003.1 standard. Resolution: 1 microsecond. Note: This timestamp must appear as the first item in an RVT LDS pack when used.
03	06 0E 2B 34 01 01 01 01 0E 01 01 01 0A 00 00 00	Platform True Airspeed	Meters / Second	uint16	2	True airspeed (TAS) of platform. Indicated Airspeed adjusted for temperature and altitude. 1 m/s = 1.94384449 knots. Resolution: 1 meter/second.
04	06 0E 2B 34 01 01 01 01 0E 01 01 01 0B 00 00 00	Platform Indicated Airspeed	Meters / Second	uint16	2	Indicated airspeed (IAS) of platform. Derived from Pitot tube and static pressure sensors. 1 m/s = 1.94384449 knots. Resolution: 1 meter/second.
05	06 0E 2B 34 01 01 01 01 0E 01 01 03 14 00 00 00	Telemetry Accuracy Indicator	None	uint8	1	Reserved for future use
06	06 0E 2B 34 01 01 01 01 0E 01 01 03 15 00 00 00	Frag Circle Radius	Meters	uint16	2	Size of fragmentation circle selected by the aircrew. Resolution: 1 meter
07	06 0E 2B 34 01 01 01 01 01 04 07 02 00 00 00 00	Frame Code	None	uint32	4	Range is from 0 to 4,294,967,296. Counter runs at 60 Hz
08	06 0E 2B 34 01 01 01 01 0E 01 02 03 03 00 00 00	UAS LDS Version Number	Number	uint8	1	Version number of the LDS document used to generate a source of LDS KLV metadata. 0 is pre-release, initial release (0806.0), or test data. 1..255 corresponds to document revisions 1 thru 255.
09	06 0E 2B 34 01 01 01 01 0E 01 01 01 19 00 00 00	Video Datarate	bps or Hz	uint32	4	Video data rate (Digital only), or Analog FM

RVT LDS						
Key Value (hex)		Key Name	Units	Format	Length in Bytes	Notes
10	06 0E 2B 34 01 01 01 03 04 01 0B 01 00 00 00 00	Digital Video File Format	String	ISO7	Max. 127	Video Compression being used. Maximum 127 characters. Examples: MPEG2 MPEG4 H.264 Analog FM (non compressed) As this list is not exhaustive, other values or variants are also acceptable.
11	06 0E 2B 34 02 0B 01 01 0E 01 03 01 0F 00 00 00	User Defined Data Packlet	Varies	N/A	V	Local set key to include user defined data items within the RVT KLV Dictionary. Use the values of the items specified within the User Defined Data Packlet. The length field is the size of all POI items to be packaged within this tag.
12	06 0E 2B 34 02 0B 01 01 0E 01 03 01 0C 00 00 00	Point of Interest Local Data Set	None	N/A	V	Local set key to include POI items within RVT KLV Dictionary. Use POI local set tags within a POI packet. The length field is the size of all POI items to be packaged within this tag.
13	06 0E 2B 34 02 0B 01 01 0E 01 03 01 0D 00 00 00	Area of Interest Local Data Set	None	N/A	V	Local set key to include AOI items within RVT KLV Dictionary. Use AOI local set tags within an AOI packet. The length field is the size of all AOI items to be packaged within this tag.
14	06 0E 2B 34 01 01 01 01 0E 01 01 03 0A 00 00 00	MGRS Zone	None	uint8	1	AIRCRAFT: First two characters of Aircraft MGRS coordinates, UTM zone 01 through 60
15	06 0E 2B 34 01 01 01 01 0E 01 01 03 0B 00 00 00	MGRS Latitude Band and Grid Square	String	ISO7	3	AIRCRAFT: Third, fourth and fifth characters of Aircraft MGRS coordinates. Third character is the alpha code for the latitude band (C through X, omitting I and O). Fourth and fifth characters are the 2-digit alpha code for the grid square designator (WGS 84).
16	06 0E 2B 34 01 01 01 01 0E 01 01 03 0C 00 00 00	MGRS Easting	Meters	uint24	3	AIRCRAFT: Sixth through tenth character of Aircraft MGRS coordinates. Range is from 0 to 99,999 representing the 5-digit Easting value in meters. Resolution: 1 meter
17	06 0E 2B 34 01 01 01 01 0E 01 01 03 0D 00 00 00	MGRS Northing	Meters	uint24	3	AIRCRAFT: Eleventh through fifteenth character of Aircraft MGRS coordinates. Range is from 0 to 99,999 representing the 5-digit Northing value in meters. Resolution: 1 meter
18	06 0E 2B 34 01 01 01 01 0E 01 01 03 0A 00 00 00	MGRS Zone	None	uint8	1	FRAME CENTER: First two characters of Frame Center MGRS coordinates, UTM zone 01 through 60

RVT LDS						
Key Value (hex)		Key Name	Units	Format	Length in Bytes	Notes
19	06 0E 2B 34 01 01 01 01 0E 01 01 03 0B 00 00 00	MGRS Latitude Band and Grid Square	String	ISO7	3	FRAME CENTER: Third, fourth and fifth characters of Frame Center MGRS coordinates. Third character is the alpha code for the latitude band (C through X, omitting I and O). Fourth and fifth characters are the 2-digit alpha code for the grid square designator (WGS 84).
20	06 0E 2B 34 01 01 01 01 0E 01 01 03 0C 00 00 00	MGRS Easting	Meters	uint24	3	FRAME CENTER: Sixth through tenth character of Frame Center MGRS coordinates. Range is from 0 to 99,999 representing the 5-digit Easting value in meters. Resolution: 1 meter
21	06 0E 2B 34 01 01 01 01 0E 01 01 03 0D 00 00 00	MGRS Northing	Meters	uint24	3	FRAME CENTER: Eleventh through fifteenth character of Frame Center MGRS coordinates. Range is from 0 to 99,999 representing the 5-digit Northing value in meters. Resolution: 1 meter

Table 6-2: Point of Interest (POI) Local Data Set

Point of Interest Local Data Set						
Key Value (hex)		Key Name	Units	Format	Length in Bytes	Notes
06 0E 2B 34 02 0B 01 01 0E 01 03 01 0C 00 00 00		Point of Interest Local Data Set	None	N/A	Variable	This is the Universal Key for the Point of Interest Local Data Set
Item	Key Value (hex)	Key Name	Units	Format	Length in Bytes	Notes
01	06 0E 2B 34 01 01 01 01 0E 01 01 03 16 00 00 00	POI/AOI Number	Number	uint16	2	POI Number ** REQUIRED when sending a POI **
02	06 0E 2B 34 01 01 01 01 0E 01 01 03 17 00 00 00	POI Latitude	Degrees	int32	4	POI Latitude. Based on WGS84 ellipsoid. Map -(2^31-1)..(2^31-1) to +/- 90. Use -(2^31) as an "error" indicator. -(2^31) = 0x80000000. Resolution: ~42 nano degrees. ** REQUIRED when sending a POI **
03	06 0E 2B 34 01 01 01 01 0E 01 01 03 18 00 00 00	POI Longitude	Degrees	int32	4	POI Longitude. Based on WGS84 ellipsoid. Map -(2^31-1)..(2^31-1) to +/- 180. Use -(2^31) as an "error" indicator. -(2^31) = 0x80000000. Resolution: ~84 nano degrees. ** REQUIRED when sending a POI **
04	06 0E 2B 34 01 01 01 01 0E 01 01 03 19 00 00 00	POI Altitude	Meters	uint16	2	Altitude of POI as measured from Mean Sea Level (MSL). Map 0..(2^16-1) to -900..19000 meters. 1 meter = 3.2808399 feet. Resolution: ~0.3 meters.
05	06 0E 2B 34 01 01 01 01 0E 01 01 03 1A 00 00 00	POI/AOI Type	None	int8	1	Target Identifier: 1="Friendly", 2="Hostile", 3="Target", or 4="Unknown"
06	06 0E 2B 34 01 01 01 01 0E 01 01 03 1B 00 00 00	POI/AOI Text	String	ISO7	Max. 2048	User Defined String.
07	06 0E 2B 34 01 01 01 01 0E 01 01 03 1C 00 00 00	POI Source Icon	String	ISO7	Max. 127	Per MIL-STD-2525B. Maximum 127 characters. Icon used in FalconView.
08	06 0E 2B 34 01 01 01 01 0E 01 01 03 1D 00 00 00	POI/AOI Source ID	String	ISO7	Max. 255	User Defined String.
09	06 0E 2B 34 01 01 01 01 0E 01 01 03 1E 00 00 00	POI/AOI Label	String	ISO7	16	User Defined String

Table 6-3: Area of Interest (AOI) Local Data Set

Area of Interest Local Data Set						
Key Value (hex)		Key Name	Units	Format	Length in Bytes	Notes
06 0E 2B 34 02 0B 01 01 0E 01 03 01 0D 00 00 00		Area of Interest Local Data Set	None	N/A	Variable	This is the Universal Key for the Area of Interest Local Data Set
Tag ID	Key Value (hex)	Key Name	Units	Format	Length in Bytes	Notes
01	06 0E 2B 34 01 01 01 01 0E 01 01 03 16 00 00 00	POI/AOI Number	Number	uint16	2	AOI Number ** REQUIRED when sending an AOI **
02	06 0E 2B 34 01 01 01 03 07 01 02 01 03 07 01 00	Upper Left Lat	Degrees	int32	4	NW corner of AOI. Based on WGS84 ellipsoid. Map -(2^31-1)..(2^31-1) to +/- 90. Use -(2^31) as an "error" indicator. -(2^31) = 0x80000000. Resolution: ~42 nano degrees. ** REQUIRED when sending an AOI **
03	06 0E 2B 34 01 01 01 03 07 01 02 01 03 0B 01 00	Upper Left Long	Degrees	int32	4	NW corner of AOI. Based on WGS84 ellipsoid. Map -(2^31-1)..(2^31-1) to +/- 180. Use -(2^31) as an "error" indicator. -(2^31) = 0x80000000. Resolution: ~84 nano degrees. ** REQUIRED when sending an AOI **
04	06 0E 2B 34 01 01 01 03 07 01 02 01 03 09 01 00	Lower Right Lat	Degrees	int32	4	SE corner of AOI. Based on WGS84 ellipsoid. Map -(2^31-1)..(2^31-1) to +/- 90. Use -(2^31) as an "error" indicator. -(2^31) = 0x80000000. Resolution: ~42 nano degrees. ** REQUIRED when sending an AOI **
05	06 0E 2B 34 01 01 01 03 07 01 02 01 03 0D 01 00	Lower Right Long	Degrees	int32	4	SE corner of AOI. Based on WGS84 ellipsoid. Map -(2^31-1)..(2^31-1) to +/- 180. Use -(2^31) as an "error" indicator. -(2^31) = 0x80000000. Resolution: ~84 nano degrees. ** REQUIRED when sending an AOI **
06	06 0E 2B 34 01 01 01 01 0E 01 01 03 1A 00 00 00	POI/AOI Type	None	int8	1	Target Identifier: 1="Friendly", 2="Hostile", 3="Reserved", or 4="Unknown" ** REQUIRED when sending an AOI **
07	06 0E 2B 34 01 01 01 01 0E 01 01 03 1B 00 00 00	POI/AOI Text	String	ISO7	Max. 2048	User Defined String.
08	06 0E 2B 34 01 01 01 01 0E 01 01 03 1D 00 00 00	POI/AOI Source ID	String	ISO7	Max. 255	User Defined String.
09	06 0E 2B 34 01 01 01 01 0E 01 01 03 1E 00 00 00	POI/AOI Label	String	ISO7	16	User Defined String.

Table 6-4: User Defined Data Packlet

User Defined Data Packlet						
Key Value (hex)		Key Name	Units	Format	Length in Bytes	Notes
06 0E 2B 34 02 0B 01 01 0F 01 03 01 0E 00 00 00		User Defined Data Packlet	None	N/A	Variable	This is the Universal Key for the User Defined Data Packlet
Item	Key Value (hex)	Key Name	Units	Format	Length in Bytes	Notes
01	06 0E 2B 34 01 01 01 01 0E 01 02 03 11 00 00 00	Numeric ID for Data Type	N/A	uint8	1	Numeric identifier with data type. Bit Ordering msb first: 87654321 bits 8 & 7 sets the data type. = 00 for strings = 01 for INT = 10 for UINT = 11 for Experimental. bits 1 to 6 is the integer numeric ID for the user defined data ranging from 0 to 63 (64 possible user data items for each type) ** REQUIRED when sending User Defined Data **
02	06 0E 2B 34 01 01 01 01 0E 01 02 03 12 00 00 00	User Data	N/A	N/A	V	User Data. Data type defined in byte 1 of this packet with variant (uint16, uint32, etc) extracted from the overall pack length. ** REQUIRED when sending User Defined Data **

7 Glossary of Acronyms and Symbols

AOI	Area of Interest
BER	Basic Encoding Rules
KLV	Key-Length-Value
LDS	Local Data Set
MISB	Motion Imagery Standards Board
MSB	Most Significant Byte
msb	Most Significant Bit
OSRVT	One System Remote Video Terminal
POI	Point of Interest
ROVER	Remote Operations Video Enhanced Receiver
RVT	Remote Video Terminal
UINT	Unsigned Integer

8 Appendix: Calculating the RVT LDS CRC

A Cyclic Redundancy Check (CRC) is a type of hash function used to produce a checksum. The checksum is used to detect errors after transmission or storage. A CRC is computed and appended before transmission or storage, and verified afterwards by recipient to confirm that no changes occurred on transit. The CRC used is the CRC-32-ISO/IEC 13818-1:2000 (MPEG2).

Below is an example of C++ code necessary to calculate the CRC-32 on a block of data:

```
class CRC32MPEG
{
private: // CONSTANTS

  static const unsigned long DEFAULT_POLYNOMIAL = 0x04c11db7L;
  static const unsigned long CRC_ACCUM_INIT = 0xFFFFFFFF;

public: // FUNCTIONS

  CRC32MPEG(void);
  unsigned long Update(char *data_blk_ptr, int data_blk_size);
  inline void Init(void) { m_crcAccum = CRC_ACCUM_INIT; }

private: // VARIABLES

  unsigned long m_crcAccum;
  static const unsigned long m_crcTable[256];
};
```

```
/* MPEG CRC-32 Table */
const unsigned long CRC32MPEG::m_crcTable[256] = {
  0x00000000, 0x04C11DB7, 0x09823B6E, 0x0D4326D9, 0x130476DC, 0x17C56B6B,
  0x1A864DB2, 0x1E475005, 0x2608EDB8, 0x22C9F00F, 0x2F8AD6D6, 0x2B4BCB61,
  0x350C9B64, 0x31CD86D3, 0x3C8EA00A, 0x384FBDBD, 0x4C11DB70, 0x48D0C6C7,
  0x4593E01E, 0x4152FDA9, 0x5F15ADAC, 0x5BD4B01B, 0x569796C2, 0x52568B75,
  0x6A1936C8, 0x6ED82B7F, 0x639B0DA6, 0x675A1011, 0x791D4014, 0x7DDC5DA3,
  0x709F7B7A, 0x745E66CD, 0x9823B6E0, 0x9CE2AB57, 0x91A18D8E, 0x95609039,
  0x8B27C03C, 0x8FE6DD8B, 0x82A5FB52, 0x8664E6E5, 0xBE2B5B58, 0xBAEA46EF,
  0xB7A96036, 0xB3687D81, 0xAD2F2D84, 0xA9EE3033, 0xA4AD16EA, 0xA06C0B5D,
  0xD4326D90, 0xD0F37027, 0xDDB056FE, 0xD9714B49, 0xC7361B4C, 0xC3F706FB,
  0xCEB42022, 0xCA753D95, 0xF23A8028, 0xF6FB9D9F, 0xFBB8BB46, 0xFF79A6F1,
  0xE13EF6F4, 0xE5FFEB43, 0xE8BCCD9A, 0xEC7DD02D, 0x34867077, 0x30476DC0,
  0x3D044B19, 0x39C556AE, 0x278206AB, 0x23431B1C, 0x2E003DC5, 0x2AC12072,
  0x128E9DCF, 0x164F8078, 0x1B0CA6A1, 0x1FCDBB16, 0x018AEB13, 0x054BF6A4,
  0x0808D07D, 0x0CC9CDCA, 0x7897AB07, 0x7C56B6B0, 0x71159069, 0x75D48DDE,
  0x6B93DDDB, 0x6F52C06C, 0x6211E6B5, 0x66D0FB02, 0x5E9F46BF, 0x5A5E5B08,
  0x571D7DD1, 0x53DC6066, 0x4D9B3063, 0x495A2DD4, 0x44190B0D, 0x40D816BA,
  0xACA5C697, 0xA864DB20, 0xA527FDF9, 0xA1E6E04E, 0xBFA1B04B, 0xBB60ADFC,
  0xB6238B25, 0xB2E29692, 0x8AAD2B2F, 0x8E6C3698, 0x832F1041, 0x87EE0DF6,
  0x99A95DF3, 0x9D684044, 0x902B669D, 0x94EA7B2A, 0xE0B41DE7, 0xE4750050,
  0xE9362689, 0xEDF73B3E, 0xF3B06B3B, 0xF771768C, 0xFA325055, 0xFEF34DE2,
  0xC6BCF05F, 0xC27DEDE8, 0xCF3ECB31, 0xCBFFD686, 0xD5B88683, 0xD1799B34,
  0xDC3ABDED, 0xD8FBA05A, 0x690CE0EE, 0x6DCDFD59, 0x608EDB80, 0x644FC637,
  0x7A089632, 0x7EC98B85, 0x738AAD5C, 0x774BB0EB, 0x4F040D56, 0x4BC510E1,
  0x46863638, 0x42472B8F, 0x5C007B8A, 0x58C1663D, 0x558240E4, 0x51435D53,
  0x251D3B9E, 0x21DC2629, 0x2C9F00F0, 0x285E1D47, 0x36194D42, 0x32D850F5,
  0x3F9B762C, 0x3B5A6B9B, 0x0315D626, 0x07D4CB91, 0x0A97ED48, 0x0E56F0FF,
  0x1011A0FA, 0x14D0BD4D, 0x19939B94, 0x1D528623, 0xF12F560E, 0xF5EE4BB9,
  0xF8AD6D60, 0xFC6C70D7, 0xE22B20D2, 0xE6EA3D65, 0xEBA91BBC, 0xEF68060B,
  0xD727BBB6, 0xD3E6A601, 0xDEA580D8, 0xDA649D6F, 0xC423CD6A, 0xC0E2D0DD,
  0xCDA1F604, 0xC960EBB3, 0xBD3E8D7E, 0xB9FF90C9, 0xB4BCB610, 0xB07DABA7,
  0xAE3AFBA2, 0xAAFBE615, 0xA7B8C0CC, 0xA379DD7B, 0x9B3660C6, 0x9FF77D71,
  0x92B45BA8, 0x9675461F, 0x8832161A, 0x8CF30BAD, 0x81B02D74, 0x857130C3,
  0x5D8A9099, 0x594B8D2E, 0x5408ABF7, 0x50C9B640, 0x4E8EE645, 0x4A4FFBF2,
  0x470CDD2B, 0x43CDC09C, 0x7B827D21, 0x7F436096, 0x7200464F, 0x76C15BF8,
  0x68860BFD, 0x6C47164A, 0x61043093, 0x65C52D24, 0x119B4BE9, 0x155A565E,
  0x18197087, 0x1CD86D30, 0x029F3D35, 0x065E2082, 0x0B1D065B, 0x0FDC1BEC,
  0x3793A651, 0x3352BBE6, 0x3E119D3F, 0x3AD08088, 0x2497D08D, 0x2056CD3A,
  0x2D15EBE3, 0x29D4F654, 0xC5A92679, 0xC1683BCE, 0xCC2B1D17, 0xC8EA00A0,
  0xD6AD50A5, 0xD26C4D12, 0xDF2F6BCB, 0xDBEE767C, 0xE3A1CBC1, 0xE760D676,
  0xEA23F0AF, 0xEEE2ED18, 0xF0A5BD1D, 0xF464A0AA, 0xF9278673, 0xFDE69BC4,
  0x89B8FD09, 0x8D79E0BE, 0x803AC667, 0x84FBDBD0, 0x9ABC8BD5, 0x9E7D9662,
  0x933EB0BB, 0x97FFAD0C, 0xAFB010B1, 0xAB710D06, 0xA6322BDF, 0xA2F33668,
  0xBCB4666D, 0xB8757BDA, 0xB5365D03, 0xB1F740B4
};

CRC32MPEG::CRC32MPEG(void)
: m_crcAccum(CRC_ACCUM_INIT) {
}
unsigned long CRC32MPEG::Update(char *data_blk_ptr, int data_blk_size) {
  for(int j = 0; j < data_blk_size; j++) {
    int i = ((int)(m_crcAccum >> 24) ^ *data_blk_ptr++) & 0xff;
    m_crcAccum = (m_crcAccum << 8) ^ m_crcTable[i];
  }
  return m_crcAccum;
}
```